Q76495.APP.CWB
July 10, 2003

INDEX STRUCTURE OF METADATA , METHOD FOR PROVIDING INDICES OF METADATA, AND METADATA SEARCHING METHOD AND APPARATUS USING THE INDICES OF METADATA

BACKGROUND OF THE INVENTION

1. Field of the Invention

[01] The present invention relates to an index structure of metadata provided for searching for information on contents and a method for providing indices of the metadata, and a method and an apparatus for searching for the metadata using the index structure of the metadata. More particularly, the present invention relates to an index structure of metadata provided for searching for information on contents and a method for providing indices of the metadata, and a method and an apparatus for searching for the metadata using the indices of metadata, the metadata containing multi-keys with which information on contents can be more efficiently searched when XML metadata on digital contents defined by TV-Anytime Forum (hereinafter referred to as "TVA metadata") is divided into fragments as an independent unit and transmitted on a fragment basis. The present application is based on Korean Patent Application Nos. 2002-43097 and 2002-62923, which are incorporated herein by reference.

2. Description of the Prior Art

[02] The TV-Anytime Forum is a private standardization organization established in September 1999 with the purpose of developing standards for providing audiovisual related services in a user-friendly environment such as a personal digital recorder (PDR) having a high volume personal storage device. Specifically, the aim of the services is to enable all the users to view and listen to various types of programs (such as conventional broadcasting services, online interactive services and the like) at a desired time and in a desired manner based on the personal storage device.

[03] The TV-Anytime Forum has operated Working Groups for business models, system/transmission interfaces/contents referencing, descriptions, metadata, rights management and protection and the like, in order to establish standardization. With respect to the metadata concerned in the present invention, "1st Draft of Metadata Specification SP003v1.3" up to June 2002 has been published.

[04] A configuration of the PDR will be briefly described with reference to FIG. 1. The PDR 100 receives video/audio signals and metadata via a variety of networks such as sky waves, satellite waves, internet networks and the like from a provider 200 for providing video/audio signals, collects viewing and listening patterns, and personal tastes of users, if necessary, and transmits them to the provider 200 for providing the video/audio signals. The PDR 100 comprises a high volume storage device for storing therein the received

video/audio signals and metadata. The PDR 100 further comprises software for storage and reproduction of the video/audio signals, and an electronic program guide (EPG) application for retrieving and displaying metadata for the video/audio signals. The user ascertains the metadata for the video/audio data, i.e., titles of the programs, program reproduction times and the like, through a grid guide screen of the EPG application shown in FIG. 2, selects a desired program, and receives it via the network in real time or reproduces the video/audio data previously stored in the high volume storage device.

[05] The metadata refer to data describing contents such as titles and synopses of programs, and are defined as "data about data." In the TVA metadata specifications of the TV-Anytime Forum, its structure is defined by use of XML schema language (see XML 1.0 of W3C), the standard by the W3C (a consortium for promoting standards for the XML), and the semantics and attributes of the respective metadata elements are also defined. The TVA metadata relevant to broadcasting contents are configured with an XML document having a root node, "TVAMain (300)" as shown in FIG. 3. The TVA metadata relevant to programs are configured with, for example, nodes such as ProgramInformation Table, GroupInformation Table, ProgramLocation Table, ServiceInformation Table and the like, under the node of "ProgramDescription."

[06] In the TV-Anytime Forum, the TVA metadata are transmitted on a fragment basis as an independent unit in order to transmit a large volume of

TVA metadata in a stream format. The concept of fragments will be briefly described with reference to FIG. 4. The fragments are obtained by dividing the TVA metadata configured with the XML documents shown in FIG. 3 into predetermined tree structures. For example, where the entire TVA metadata are divided into a tree structure (fragment TVAMain) including an upper node of "TVAMain" and predetermined child nodes under this upper node, a tree structure (fragment ProgramInformation) including an upper node of ProgramInformation Table and child nodes under this upper node, a tree structure (fragment BroadcastEvent) including an upper node of the BroadcastEvent Information and child nodes under this upper node, each of the divided tree structures becomes a fragment. The fragments can be transmitted independently of the other fragments, and the fragments can be accessed individually.

[07] For individual access to the fragments, it is necessary to know a node referenced by a transmitted TVA metadata fragment, i.e., a node corresponding to the upper node of the TVA metadata fragment, in the entire metadata tree structure, and to describe relative paths in the TVA metadata fragments of keys contained in the transmitted TVA metadata fragment. To this end, XPath, which is a syntax for describing a path to one or more nodes in an XML document defined by W3C, is used. The term 'key' refers to a specific field of the metadata used for indexing, and also means child nodes of a node referenced by a fragment. Fields (for search conditions) input by the user, such as 'Service ID' and 'Published Time' correspond to the keys.

[08] In order to provide efficient search for and access to fragments, an index structure for the keys included in the metadata fragments is additionally required, and information on the index structure, i.e., index information, is also transmitted independently of the metadata fragments.

[09] Under the environment provided by the TV-Anytime Forum, if a user desires to retrieve information on a program meeting a predetermined Published Time condition, the index information transmitted thereto independently of the fragments is utilized to identify the location (identifier) of a metadata fragment meeting a desired Published Time condition and an access to the relevant metadata fragment is then made based on the location (identifier), so as to extract metadata meeting the Published Time condition.

[10] TV-Anytime Specification TV145, J.P. Evain, "1st Draft of Metadata Specification SP003v1.3", TV-Anytime Forum 17th meeting, Montreal, Canada, June 2002; hereinafter, referred to as "Single key index art reference" proposes a single key index structure for a metadata fragment index.

[11] Note that the term "single key" is used herein to distinguish it from a concept of the term "multi-key" in an embodiment of the present invention to be described later. A multi-key index structure according to an embodiment of the present invention enables the user to access metadata for a plurality of keys, using a plurality of the keys simultaneously, but a single key index

structure of the prior art allows only one single key to be used for an access to the metadata.

[12] The notion of a container defined by the TV-Anytime Forum will be described prior to describing the index structure.

[13] The TV-Anytime Forum defines a container as a top-level storage to which all the data covering the aforementioned index information and the metadata fragments are transmitted, which is called a type of top-level transmission. Describing the container briefly, each container comprises a plurality of sections, each storing therein the index information or the metadata fragments. The container can be classified into an index container and a data container according to the information carried thereby: the index container carries index information sections such as a key index list (key_index_list) section, a key index (key_index) section, a sub key index (sub_key_index) section, a string repository (string_repository) section and a fragment data repository (fragment_data_repository) section, whereas a data container carries metadata fragment sections such as an elements table (elements_table) section, a string repository (string_repository) section and a fragment data repository (fragment_data_repository) section. The above classification is done based on the contents of the information included in the containers. Both the index container and the data container are identical in configuration.

[14] Referring to the container defined by the TV-Anytime Forum as illustrated in FIG. 5, the container comprises a container identifier (container_id) data field (not shown) and a large number of sections. In each section, the contents stored in 'section_body' are identified according to an encoded value in 'section_id'. For example, a section 10 of which the encoded value in 'section_id' is '0X0004' is identified as a key index list (key_index_list) section, a section 20 of which the encoded value in 'section_id' is '0X0005' is identified as a key index (key_index) section, a section 30 of which the encoded value in 'section id' is '0X0006' is identified as a sub key index (sub_key_index) section, a section 40 of which the encoded value in 'section id' is '0X0001' is identified as an element table (element_table) section, and a section 50 of which the encoded value in 'section id' is '0X0003' is identified as a fragment data repository (fragment_data_repository) section.

[15] The TVA metadata fragments are stored in the fragment data repository (fragment_data_repository) section 50 of the data container and then transmitted. The identifier information (handle_value) for the TVA metadata fragments in the data container is included in the element table section 40 of the data container.

[16] In conclusion, the TVA metadata fragment is uniquely identified by the container identifier information (container_id) and the metadata fragment

identifier information (handle_value) of the container that includes the TVA metadata fragment.

[17] The single key index art reference described above proposes the single key index structure for indexing the TVA metadata fragments stored in the aforementioned data container, i.e., a structure composed of the key index list (key_index_list) section 10, the key index (key_index) section 20, and the sub key index (sub_key_index) section 30. Since the syntax of the structure is described in detail in the single key index reference described above, the detailed description thereof will be omitted. Hereinafter, the structure will be described with reference to FIG. 6 that illustrates the structure by segments of the index information.

[18] The key index list (key_index_list) section 10 defined in the single key index structure provides a list of all the single keys transmitted. The list includes single key information defining each single key and identification information on the key index (key_index) section 20 to be described later. The single key information comprises (1) location information of the metadata fragment relevant to the single key, (2) location information of the single key within the metadata fragment and identification information for the key index (key_index) section 20 which will be set forth later. The location information of the metadata fragment is expressed in XPath (fragment_xpath_ptr) in the TVA. The location information of the single key is expressed in XPath

(key_xpath_ptr) for the relative path within the relevant fragment of the node used as the single key in the TVA.

[19] The XPath of the metadata fragment is a path to the root node of the TVA metadata XML document, i.e., an absolute path, and the XPath of the nodes used as the single keys, i.e., the XPath of the single keys, represents a relative path of the single key for the relevant metadata fragment. The XPath for the metadata fragment and the XPath for the single key are stored in a 'fragment_xpath_ptr' segment 11 and a 'key_xpath_ptr' segment 12, respectively.

[20] Furthermore, the key index list (key_index_list) section 10 includes the identification information on the key index (key_index) section 20 of each single key to be described later (i.e., the container identifier information (container_id) of the container storing therein the key index (key_index) section 20 and the key index identifier information). The container identifier information and the key index identifier information are stored in an 'index_container' segment of the key index list (key_index_list) section 10 and a 'key_index_identifier' segment, respectively, and then transmitted.

[21] The key index (key_index) section 20 defined in the single key index structure provides a list of all the sub key index (sub_key_index) sections 30 to be described later. The list includes information representing the ranges of values of the key included in the respective sub key index (sub_key_index) sections 30, i.e., the highest value of the key among the values of the key

within each sub key index (sub_key_index) section 30 (hereinafter referred to as 'representative key value'), and identification information on the sub key index (sub_key_index) section 30 relevant to each representative key value (i.e., the container identifier information (container_id) of the container storing therein the sub key index (sub_key_index) section, and the sub key index identifier information).

[22] Accordingly, the key index section (key_index) 20 includes a 'key_index_identifier' segment for storing therein the key index identifier information defined in the key index list (key_index_list) section 10, a 'high_key_value' segment 13 for storing therein the representative key values of the respective sub key index (sub_key_index) sections 30, the container identifier information (container_id) of the container in which the sub key index (sub_key_index) section 30 is stored, a 'sub_index_container' segment for storing respective sub key index identifier information and a 'sub_index_identifier' segment.The sub key index (sub_key_index) section 30 defined in the single key index structure provides a list of the values of the key included in the relevant sub key index (sub_key_index) section 30. The list includes the values of the key included in the relevant sub key index (sub_key_index) section 30 and the identification information on the metadata fragments having the values of the key (i.e., the container identifier information (container_id) of the containers storing the metadata fragments and the identifier information (handle_value) of the metadata fragments).

[23] Accordingly, the sub key index (sub_key_index) section 30 includes a 'sub_index_identifier' segment for storing therein the sub key index identifier information defined in the key index (key_index) section 20, a 'key_value' segment 14 for storing therein the values of the key, a 'target_container' segment for storing therein the container identifier information (container_id) of the containers in which the metadata fragments are stored, and a 'target_handle' segment for storing therein the fragment data identifier information (handle_value). The single key index structure may be more easily understood by referring to FIG. 7 illustrating the index information.

[24] FIGS. 7a and 7b show the key index list (key_index_list) section including single keys relevant to the Service Id, the Published Time and the Published Duration. The upper node of the metadata fragment including the single keys relevant to the Service Id, the Published Time and the Published Duration is 'BroadcastEvent' 310 as shown in FIG. 3, identified by a shaded block. Accordingly, the XPath '/TVAMain/ProgramDescription/Program-Location Table/BroadcastEvent' for the 'BroadcastEvent' fragment is stored in the 'fragment_xpath_ptr' segment 11a, and the XPaths to the single keys of the Service Id, the Published Time and the Published Duration for the 'BroadcastEvent' fragment, i.e., '@ServiceId' (311a in FIG. 3), 'EventDescription/PublishedTime' (311b in FIG. 3) and 'EventDescription/ PublishedDuration' (311c in FIG. 3) are stored in the 'key_xpath_ptr' segment 12a.

[25] Illustratively, FIG. 7a shows a key index (key_index) section 20a and a sub key index (sub_key_index) section 30a for the Service Id (XPath of the single key: @ServiceId) among the key index list (key_index_list) sections. FIG. 7b shows a key index (key_index) section 20b and a sub key index (sub_key_index) section 30b for the Published Time (XPath of the single key: EventDescription/PublishedTime).

[26] This single key index structure is disadvantageous in that it is inefficient to perform a compound condition search, i.e., a search by one or more search conditions, since it can only support a single key search, i.e., an index search using a key corresponding to a specific field of the metadata fragment according to the TV-Anytime specification. For example, in order to display a list of broadcast programs on the grid guide screen as shown in FIG. 2, search operations for two fields, i.e., the Service Id and the Published Time, are required.

[27] In order to explain the compound condition search using a conventional single key index structure, a case where a list of programs of which a Service Id is in the range of 507 to 514 and the Published Time endures for 09:30 to 10:00 is obtained will be explained hereinafter by way of example. In the TV-Anytime metadata specification, search conditions for retrieving metadata related to the program list are expressed as follows.

- Fragment targeted for search (BroadcastEvent):

/TVAmain/ProgramDescription/ProgramLocationTable/BroadcastEven

t,

- List of search conditions:

507 <= ServiceId <= 514

09:30 <= EventDescription/PublishedTime <= 10:00.

[28] In the conventional single key index structure, two methods are available for obtaining fragments meeting the designated search conditions. The methods will be described in detail with reference to FIGS. 8a and 8b.

(1) First search method using the single key index

[29] In the first method, as shown in FIG. 8a, sets of fragments as intermediate results meeting respective conditions are independently searched by use of respective single keys for the ServiceId and the EventDescription/PublishedTime. Thereafter, fragments common in both sets of the independently searched fragments are obtained, among which a final resultant set of fragments meeting the conditions is obtained.

[30] Hereinafter, this method will be described in detail with reference to FIGS. 7a and 8a.

[31] First, single key information and the value of the single key required for the Service Id search is designated (S11). The single key information comprises XPath of the search target metadata fragment as location information of the search target metadata fragment, and XPath of the single key as location information of the single key within the metadata fragment.

[32] - XPath of the metadata fragment:

[33] /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent,

[34] - XPath of the Service Id: @ServiceId,

[35] - Value of key of the Service Id: 507<= ServiceId <= 514.

[36] Subsequently, the single key corresponding to the XPath 11a of the fragment and the XPath 12a of the Service Id is retrieved from a key index list (key_index_list) section 10a, and identification information on a key index (key_index) section 20a is extracted. On this basis, representative key values of '509' 13a and '519' 13a, i.e., representative key values which indicate a range (500-509, 510-519) of values of the key in which values of the key (507-514) of the single keys to be searched are included, are retrieved from the key index (key_index) section 20a having the extracted identification information. Then, identification information on sub key index (sub_key_index) sections 14a having the values of the key (500-509, 510-519) related to the representative key values '509' and '519' is extracted. The identification information of the metadata fragment (the container identifier information (container_id) and the fragment data identifier information (handle_value) stored in a 'target_container' segment and a 'target_handle' segment, respectively) corresponding to the values of key of 507-514 is

extracted from the sub key index (sub_key_index) sections 14a having the extracted identification information, and the relevant metadata fragment is extracted by use of the extracted identification information (S12, S14).

[37] For searching the Published Time as an example, the single key information, i.e., XPath information of the search target metadata fragment and XPath information of the single key, and the value of the single key are expressed as follows.

[38] - XPath of the fragment:

[39] /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEve nt,

[40] - XPath of the Published Time: EventDescription/PublishedTime,

[41] - Value of the key of the Published Time: 09:30<= EventDescription/ PublishedTime <=10:00. Metadata fragments corresponding to the values of key of 09:30-10:00 are extracted through the substantially same steps as in the Service Id search (S13, S15).

[42] The intersection between the extracted metadata fragments for the Service Id and the Published Time is performed, and metadata of common metadata fragments are provided to the grid guide screen shown in FIG. 2 as a final result (S16).

(2) Second search method using the single key index

[43] In the second method, the fragments are searched by use of only one (for example, Service Id) of the two single keys related to the search conditions as illustrated in FIG. 8b (S21-S23), and only the fragments of which the Published Time as another search condition is between 09:30 and 10:00 are selected from the searched fragments (S24).

[44] Since intermediate resultant fragments obtained through the search using the respective single keys is usually very large in number, these search methods using the single key index structure are not efficient. In the first method, since all programs in the range of the relevant Service Id are obtained as a search result independently of the range of the Published Time, and programs in the relevant time range for all the Service Ids are obtained as the search result, the size of the search result may become very large. Moreover, since the calculation is also complicated in the process of combining the two intermediate search results large in size, overhead in the receiving apparatus is considerably increased. In the second method, one intermediate result should be additionally filtered by the other search condition. Consequently, the compound condition search using the single key index structure may cause heavy overhead in the receiving apparatus.

SUMMARY OF THE INVENTION

[45] The present invention is contemplated to solve the aforementioned problems. An object of the present invention is to provide a multi-key index structure of metadata useful for a compound condition search for information on contents.

[46] Another object of the present invention is to provide a method of providing indices of the metadata useful for the compound condition of information on the contents, a method of searching for the metadata using the indices of the metadata, and a searching apparatus using the same.

[47] According to an embodiment of the present invention to accomplish the above and other objects, there is provided an index structure of metadata comprising values of multi-keys, identification information of the metadata corresponding to the value of the multi-key, wherein the multi-keys are structured by combination of predetermined fields of the metadata.

[48] Preferably, the index structure further comprises a list of the multi-keys.

[49] Also preferably, the index structure further comprises a representative key value representing a predetermined range of the values of the multi-key.

[50] It is desirable that the representative key value comprises at least one of a maximum value, a minimum value or an intermediate value among the values within the predetermined range.

[51] Desirably, the metadata comprises fragments divided by a predetermined range in a tree data structure, wherein the field constituting the multi-key corresponds to any one of the information constituting the fragments.

[52] Desirably, the identification information of the metadata comprises identification information of the fragment.

[53] It is preferable that the list of the multi-keys includes location information of the fragment to which the field constituting the multi-key belong, in the data structure, and location information of the field in the fragment.

[54] Preferably, the location information is expressed in XPath.

[55] More preferably, comparison of the values of the multi-key in size serves to compare the size of the value of the multi-key by the field having a different size of value first appearing by assigning the order of priority (k1>k2>k3>... kn) to a plurality of fields constituting the multi-key (k1, k2, k3... kn) and comparing the fields in sequence, starting from the field having the highest order of priority, wherein the values of two multi-keys are determined to be of the same size where there is no field having a different size of value, and the size of the fields is determined through an arithmetic comparison where the value of the concerned field is numerical or through a lexicographical order where it is alphabetical.

[56] Desirably, the metadata has a structure of metadata as defined in TVA.

[57] According to one embodiment to accomplish these and other objects of the present invention, there is provided a method for providing an index of metadata comprising values of multi-keys, and identification information of the metadata corresponding to the values of the multi-key, wherein the multi-key is structured by combination of predetermined fields of the metadata.

[58] Preferably, the index of metadata comprises a list of the multi-keys.

[59] Desirably, the index of metadata further comprises a representative key value representing a predetermined range of values of the multi-keys.

[60] It is preferable that the representative key value comprises at least one of a maximum value, a minimum value or an intermediate value among the values within the predetermined range.

[61] It is also preferable that the metadata comprises fragments divided by a predetermined range in a tree data structure, wherein the field constituting the multi-key corresponds to any one of the information constituting the fragments.

[62] Preferably, the identification information of the metadata refers to identification information of the fragment.

[63] Also preferably, the list of the multi-keys includes location information of the fragment to which the field constituting the multi-key belong, in the data structure, and location information of the field in the fragment.

[64] Preferably, the location information is expressed in XPath.

[65] Desirably, the values of the multi-key are arranged in sequence on the basis of size according to a predetermined rule.

[66] Further desirably, comparison of the values of the multi-key in size serves to compare the size of the value of the multi-key by the field having a different size of value first appearing by assigning the order of priority (k1>k2>k3>... kn) to a plurality of fields constituting the multi-key (k1, k2, k3... kn) and comparing the fields in sequence, starting from the field having the highest order of priority, wherein the values of two multi-keys are determined to be of the same size where there is no field having a different size of value, and the size of the fields is determined through an arithmetic comparison where the value of the concerned field is numerical or through a lexicographical order where it is alphabetical.

[67] Also desirably, the metadata has a structure of metadata as defined in TVA.

[68] According to one embodiment to accomplish the present invention, there is also provided a method of searching the metadata, comprising the steps of (a) allowing a user to input search conditions; (b) searching for a value of a multi-key corresponding to the input search conditions from a metadata index; and (c) extracting the concerned metadata by use of the searched value of the multi-key.

[69] It is desirable that the multi-key is structured by combination of predetermined fields of the metadata.

[70] It is also desirable that the metadata index comprises values of the multi-key and identification information of the metadata corresponding to the values of the multi-key.

[71] Desirably, the metadata index comprises a list of the multi-keys.

[72] Also desirably, the metadata further comprises a representative key value representing a predetermined range of the values of the multi-key.

[73] Preferably, the representative key value comprises at least one of a maximum value, a minimum value or an intermediate value among the values within the predetermined range.

[74] Also preferably, the metadata comprises fragments divided by a predetermined range in a tree data structure, wherein the field constituting the multi-key corresponds to any one of the information constituting the fragments.

[75] It is preferable that the identification information of the metadata refers to identification information of the fragment.

[76] It is also preferable that the list of the multi-keys comprises location information of the fragment to which the field constituting the multi-key belong, in the data structure, and location information of the field in the fragment.

[77] Also preferably, the location information is expressed in XPath.

[78] Preferably, the metadata has a structure of metadata as defined in TVA.

[79] Further preferably, in the step of searching for a value of the multi-key, the value of the multi-key having the same size in comparison to the value of the input search conditions is searched.

[80] It is desirable that comparison of the values of the multi-key in size serves to compare the size of the value of the multi-key by the field having a different size of value first appearing by assigning the order of priority (k1>k2>k3>... kn) to a plurality of fields constituting the multi-key (k1, k2, k3... kn) and comparing the fields in sequence, starting from the field having the highest order of priority, wherein the values of two multi-keys are determined to be of the same in size where there is no field having the different size of value, and the size of the fields is determined through an arithmetic comparison where the value of the concerned field is numerical or through a lexicographical order where it is alphabetical.

[81] Desirably, the step (c) of extracting the concerned metadata comprises the steps of (c1) extracting identification information of the metadata corresponding to the value of the searched multi-key in the metadata index; and (c2) extracting the concerned metadata by use of the extracted identification information.

[82] Preferably, the step (b) of searching for the value of the multi-key comprises the steps of (b1) searching for the representative key value meeting the input search conditions; and (b2) searching for the value of the multi-key meeting the input search conditions among the values of the multi-key in the range representing the representative key value.

[83] Preferably, the step (b) of searching for the value of the multi-key comprises the steps of (b3) searching for the multi-key meeting the search conditions in the multi-key list; and (b4) searching for the multi-key meeting the search conditions as inputted among the values of the multi-key indexed by the searched multi-key.

[84] Preferably, the step (b3) of searching for the multi-key comprises the steps of (b3-1) determining location information of the fragment to which the fields of the search conditions belong in the data structure, and location information of the fields in the fragment; and (b3-2) searching for the multi-key structured with the fields having the location information corresponding to the location information determined above, from the multi-key list.

[85] Preferably, the step (b4) of searching for the multi-key meeting the search conditions as inputted among the values of the multi-key indexed by the searched multi-key comprises the steps of (b4-1) searching for the representative value meeting the input search conditions; and (b4-2) searching for the value of the multi-key meeting the input search conditions among the values of the multi-key represented by the representative key.

[86] According to one embodiment to accomplish these and other objects of the present invention, there is provided an apparatus for searching metadata, comprising an input unit allowing a user to input search conditions therewith; and a control unit searching for a value of a multi-key meeting the input search conditions in the metadata index, and extracting the concerned metadata by use of the searched value of the multi-key.

[87] Preferably, the multi-key is structured by combination of predetermined fields of the metadata.

[88] Also preferably, the metadata index comprises values of the multi-key and identification information of the metadata corresponding to the values of the multi-key.

[89] It is desirable that the metadata index comprises a list of the multi-keys.

[90] Desirably, the metadata further comprises a representative key value representing a predetermined range of the values of the multi-key.

[91] Desirably, the representative key value comprises at least one of a maximum value, a minimum value or an intermediate value among the values within the predetermined range.

[92] It is preferable that the metadata comprises fragments divided by a predetermined range in a tree data structure, wherein the field constituting the multi-key corresponds to any one of the information constituting the fragments.

[93] Preferably, the identification information of the metadata refers to identification information of the fragment.

[94] More preferably, the list of the multi-keys comprises location information of the fragment to which the field constituting the multi-key belong, in the data structure, and location information of the field in the fragment.

[95] Desirably, the location information is expressed in XPath.

[96] It is desirable that the metadata has a structure of metadata as defined in TVA.

[97] Preferably, the control unit searches, from the metadata index, the value of the multi-key that is identical in size obtained by comparison of the value for the input search conditions and the value of the multi-key.

[98] Preferably, comparison of the values of the multi-key in size serves to compare the size of the value of the multi-key by the field having a different size of value first appearing by assigning the order of priority (k1>k2>k3>... kn) to a plurality of fields constituting the multi-key (k1, k2, k3... kn) and comparing the fields in sequence, starting from the field having the highest order of priority, wherein the values of two multi-keys are determined to be of the same size where there is no field having the different size of value, and the size of the fields is determined through an arithmetic comparison where the value of the concerned field is numerical or through a lexicographical order where it is alphabetical.

[99] Preferably, the controller extracts identification information of the metadata corresponding to the searched value of the multi-key in the metadata index, and extracts the concerned metadata by use of the extracted identification information.

[100] Preferably, the controller searches for the representative key value meeting the input search conditions, and searches for the value of the multi-key meeting the input search conditions among the values of the multi-key in the range of being represented by the representative value.

[101] Preferably, the controller searches for the multi-key meeting the search conditions from the multi-key list, and searches for the value of the multi-key meeting the input search conditions among the values of the multi-key indexed by the searched multi-key.

[102] Preferably, the controller determines location information within the data structure of the fragment to which the fields of the search conditions belong, and location information of the fields within the fragment, and searches for the multi-key structured with fields having the location information corresponding to the determined location information, from the multi-key list.

[103] Preferably, the controller searches for the representative key value meeting the input search conditions, and searches for the values of the multi-key meeting the input search conditions among the values of the multi-key in the range of being represented by the representative value.

[104] Preferably, the apparatus further comprises a receiving unit receiving the metadata and an index of the metadata; a storage unit storing therein the metadata and the metadata index, and an output unit outputting a search result by the control unit.

[105] Therefore, the compound condition searches for the metadata can be more efficiently performed by using the multi-key indexing scheme.

BRIEF DESCRIPTION OF THE DRAWINGS

[106] The above and other objects and features of the present invention will become apparent from the following description of preferred embodiments given in conjunction with the accompanying drawings, in which:

[107] FIG. 1 is a schematic diagram illustrating a concept of a general PDR;

[108] FIG. 2 shows a grid guide screen in a general EPG application;

[109] FIG. 3 shows a structure of general metadata defined by the TV-Anytime Forum;

[110] FIG. 4 is a schematic diagram illustrating a concept of a general fragment defined by the TV-Anytime Forum;

[111] FIG. 5 is a schematic diagram illustrating a concept of a general container defined by the TV-Anytime Forum;

[112] FIG. 6 shows an index structure of metadata employing a conventional single key concept;

[113] FIG. 7 illustrates an index structure of metadata and a searching process using a conventional single key scheme;

[114] FIG. 8 is a diagram illustrating a searching method for metadata using the conventional single key scheme;

[115] FIG. 9 shows an index structure of metadata based on a multi-key scheme according to an embodiment of the present invention;

[116] FIG. 10 shows an index structure of metadata and a searching process using the multi-key scheme according to an embodiment of the present invention;

[117] FIG. 11 illustrates a method of providing indices of metadata according to an embodiment of the present invention;

[118] FIG. 12 is a diagram showing a method of searching for the metadata according to an embodiment of the present invention; and

[119] FIG. 13 is a schematic diagram illustrating an apparatus for searching for the metadata according to an embodiment of the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[120] Hereinafter, embodiments of an index structure of metadata provided for searching for information on contents, a method for providing indices of metadata, and a method and an apparatus for searching for the metadata using the indices of metadata will be described in detail with reference to the accompanying drawings.

[121] The embodiments will be described on the basis of TVA metadata in this specification for the sake of description; however, this will not be interpreted or comprehended in limiting the coverage of protection of the present invention.

[122] The syntax defining the multi-key index structure, i.e., a structure composed of a key index list (key_index_list) section 110, a key index (key_index) section 120, and a sub key index (sub_key_index) section 130, for indexing TVA metadata fragments transmitted and stored in a data container, as an index structure of the metadata for searching for the information on contents, will be first described, and then the multi-key index structure defined by the syntax will be described.

[123] Differently from the syntax defined in the single key index art reference, the syntax defining an index structure of metadata, that is, the multi-key index structure according to an embodiment of the present invention comprises a structure newly introduced for the multi-key indexing concept including key_descriptor(), high_key_value_descriptor() and key_value_ descriptor(), and structures of key index list (key_index_list) section, key index (key_index) section, and sub key index (sub_key_index) section are reorganized.

1. Key Index List (key_index_list) section

[124] The key index list (key_index_list) section provides a list of all the transmitted multi-keys. In each key index list (key_index_list) structure, key_descriptor() is included so as to enable multi-key indexing, as shown in Table 1.

Table 1

Syntax	No. of Bits(changeable)
key_index_list() {	
for (j=0; j<key_index_count; j++) {	
fragment_xpath_ptr	16
key_descriptor()	
index_container	16
key_index_identifier	8
}	
}	

[125] **key_index_count:** specifies the number of all the transmitted multi-keys, i.e., the number of indices for the entire XML document.

[126] **fragment_xpath_ptr()**: describes the XPath of a target fragment of metadata to be indexed, i.e., location information of the target fragment of metadata to be indexed.

[127] **key_descriptor()**: describes a location of the XPath of the multi-key within the XPath of the target fragment group of the metadata to be indexed, i.e., location information of the multi-key within the metadata fragment, and information of the encoding indicator in each element/attribute constituting the multi-key.

[128] index_container: identifies the container in which a specified key index (key_index) section exists.

[129] key_index_identifier: identifies the key index (key_index) section within the container specified by index_container. The key index (key_index) section can be identified in a unique manner by combination of index_container and key_index_identifier.

2. Key Descriptor (key_descriptor)

[130] The multi-key is a compound key. With respect to a plurality of keys constituting the multi-key, the key_descriptor describes characteristics of the key such as the XPath of the key. Table 2 below shows the key_descriptor.

Table 2

Syntax	**No. of Bits(changeable)**
key_descriptor() {	
key_attribute_count	8
for (j=0; j<key_attribute_count; j++) {	
key_xpath_ptr	16
}	
}	

[131] key_attribute_count: specifies the number of keys that constitute a multi-key.

[132] key_xpath_ptr: indicates the path relative to fragment_xpath_ptr of the node (key) used as the multi-key.

3. Key Index (key_index) section high_key_value_descriptor() is newly introduced.

[133] In this embodiment of the present invention, high_key_value_descriptor() indicates a value of a representative key representing values of the multi-key within the concerned sub-key index (sub_key_index) section relative to the sub key index (sub_key_index) sections as many as the number (sub_index_count) of sub key index (sub_key_index) sections indicated by the key index (key_index) section. The high_key_value_descriptor() specifies the highest value among the values of the multi-key within the concerned sub key index (sub_key_index) sections. However, any reference value may be employed as far as it represents the values of the multi-key within a predetermined range of values within the concerned sub key index (sub_key_index) sections including the minimum value or the intermediate value, etc. as another embodiment of the present invention.

Table 3

Syntax	No. of Bits (changeable)
key_index() {	
key_index_identifier	8
sub_index_count	8
for (j=0; j<sub_index_count; j++) {	
high_key_value_descriptor()	16 * key_attribute_count
sub_index_container	16
sub_index_identifier	8
}	
}	

[134] **key_index_identifier**: identifies the key index (key_index) section within the container specified by index_container. The key index (key_index) section can be identified in a unique manner by combination of index_container and key_index_identifier. This is defined in the key index list (key_index_list) section.

[135] **sub_index_container**: identifies the container in which the designated sub key index (sub_key_index) exists.

[136] **sub_index_identifier**: identifies the sub key index (sub_key_index) section within the container specified by sub_index_container. The sub key index (sub_key_index) section can be identified in a unique manner by combination of sub_index_container and sub_index_identifier.

Table 4 below represents high_key_value_descriptor().

Table 4

Syntax	**No. of Bits (changeable)**
high_key_value_descriptor() {	
for (j=0; j<key_attribute_count; j++) {	
key_attribute_value	16
}	
}	

[137] **key_attribute_count**: specifies the number of keys constituting a multi-key. It is defined in the key index list (key_index_list) section.

[138] **key_attribute_value**: represents a representative key value for each key. The value encoding format is equal to the key_value of the single key indexing scheme.

[139] If high_key_value_descriptor() has a value of a multi-key, comparison of the values of the multi-key in size is performed as follows. Where the values of the multi-key are numerical, they are compared on an arithmetic basis; where values of the multi-key are alphabetical, they are ranked in lexicographical order. With respect to a multi-key (k_1, k_2, ..., k_n) which consists of keys k_1, k_2, ..., k_n, it is assumed that k_1 has the highest order of priority and k_n has the lowest order of priority. Under this assumption, considering two values of the multi-key (a_1, a_2, ..., a_n) and (b_1, b_2, ..., b_n),

[140] * the value of the multi-key $(a_1, a_2, ..., a_n)$ is larger than the value of the multi-key $(b_1, b_2, ..., b_n)$ if and only if there exists an integer i $(0 \leq i \leq n-1)$ such that for every $j(0 \leq j \leq i-1)$, $a_j = b_j$ and $a_i > b_i$.

[141] *the value of the multi-key $(a_1, a_2, ..., a_n)$ is smaller than the value of the multi-key $(b_1, b_2, ..., b_n)$ if and only if there exists an integer i $(0 \leq i \leq n-1)$ such that for every $j(0 \leq j \leq i-1)$, $a_j = b_j$ and $a_i < b_i$.

[142] * the value of the multi-key $(a_1, a_2, ..., a_n)$ is equal to the value of the multi-key $(b_1, b_2, ..., b_n)$ if and only if for every $i(1 \leq i \leq n)$, $a_i = b_i$.

4. Sub Key Index (sub_key_index) section

[143] key_value_descriptor() is newly introduced for the multi-key indexing scheme. The key_value_descriptor() represents a value of a multi-key of a target fragment indicated thereby.

Table 5

Syntax	No. of Bits (changeable)
sub_key_index() {	
sub_index_identifier	8
reference_count	8
for (j=0; j<reference_count; j++) {	
key_value_descriptor()	16 * key_attribute_count
target_container	16
target_handle	16
}	
}	

[144] **sub_index_identifier**: identifies the sub key index (sub_key_index) section within the container identified by sub_index_container. The sub key index (sub_key_index) section can be identified in a unique manner by combination of sub_index_container and sub_index_identifier. It is defined in the key index (key_index) section.

[145] **reference_count**: specifies the number of the multi-keys included in sub_key_index().

[146] **target_container**: identifies the container in which the designated metadata fragment exists.

[147] **target_handle**: identifies the metadata fragment section within the container identified by target_container. The metadata fragment section can

be identified in a unique manner by combination of target_container and target_handle.

Table 6 below shows key_value_descriptor().

Table 6

Syntax	No. of Bits (changeable)
key_value_descriptor() {	
for (j=0; j<key_attribute_count; j++) {	
key_attribute_value	16
}	
}	

[148] **key_attribute_count**: specifies the number of keys constituting a multi-key. It is defined in the key index list section.

[149] **key_attribute_value**: represents a value of each key. The format is equal to key_value in the single key index art reference.

[150] The comparison between key_value_descriptor() values is the same as the comparison between high_key_value_descriptor() values in the key index (key_index) section structure.

[151] Hereinafter, the metadata structure defined by the syntax described above will be discussed with reference to FIG. 9, which is illustrated by use of segments on the index information.

[152] The key index list (key_index_list) section 110 defined in the metadata structure provides a list of all the multi-keys transmitted. The list includes multi-key information defining each multi-key and identification information on the key index (key_index) section 120 to be described later. The multi-key information comprises (1) location information of the metadata fragment relevant to the multi-key (expressed, in the TVA, in XPath (fragment_xpath_ptr) for the metadata fragment relevant to the multi-key), and (2) location information of the multi-key within the metadata fragment (expressed, in the TVA, in XPath (key_descriptor) for the nodes used as the multi-keys, that is, a relevant path in the XPath location of the metadata fragment relevant to the nodes used as multi-keys). Like the single index structure, the XPath of the metadata fragment refers to a path for the root node of the TVA metadata XML document, i.e., an absolute path, and the XPath of the node used as the multi-key, i.e., the XPath of the multi-key, refers to a relative path of the multi-key for the metadata fragment. The XPath for the metadata fragment and the XPath for the multi-key are stored in a 'fragment_xpath_ptr' segment 111 and a 'key_descriptor' segment 112, respectively.

[153] The key index list (key_index_list) section 110 also comprises the identification information on the key index (key_index) section 120 of each multi-key to be described later (i.e., the container identifier information (container_id) of the container in which the key index (key_index) section 120

is stored and the key index identifier information). The container identifier information and the key index identifier information are respectively stored in an 'index_container' segment and a 'key_index_identifier' segment in the key index list (key_index_list) section 110 and then transmitted.

[154] The key index (key_index) section 120 defined in the multi-key index data stream structure provides a list of all the sub key index (sub_key_index) sections 130 to be described later. The list includes information on the ranges of the values of the multi-key included in the respective sub key index (sub_key_index) sections 130, i.e., the representative key value representing values of the multi-key included in each sub key index (sub_key_index) section 130 (in this embodiment, the highest value of the multi-key), and identification information for the sub key index (sub_key_index) section 130 related to each representative value (i.e., the container identifier information (container_id) of the container storing therein the sub key index (sub_key_index) section and the sub key index identifier information). The method of comparing the values of the multi-key in this embodiment is identical to that of comparing values of the multi-key described in connection with Table 4.

[155] The key index section (key_index) 120 includes a 'key_index_identifier' segment storing therein the key index identifier information defined in the key index list (key_index_list) section 110, a high_key_value_descriptor' segment 113 storing therein the representative key value of each sub key index (sub_key_index) section 130, and the identification information on the sub key index (sub_key_index) section 130 including the values of the multi-key corresponding to the range indicated by the representative key value. The identification information on the sub key index (sub_key_index) section 130a includes 'sub_index_container' segment storing therein the container identifier information (container_id) of the container in which the sub key index (sub_key_index) section 130 is stored, and a 'sub_index_identifier' segment storing therein the sub key index identifier information,

[156] The sub key index (sub_key_index) section 130 defined in the metadata structure provides a list of the values of the multi-key included in the relevant sub key index (sub_key_index) section 130. The list provides the values of the multi-key included in the relevant sub key index (sub_key_index) section 130, and the identification information on the metadata fragment having the values of the multi-key (i.e., container identifier information (container_id) of the container in which the metadata fragment is stored and the identifier information (handle_value) on the metadata fragment).

[157] The sub key index (sub_key_index) section 130 includes a 'sub_index_identifier' segment storing therein the sub key index identifier information defined in the key index (key_index) section 120, a 'key_value_descriptor' segment 114 for storing therein the values of the multi-key, and the identification information on the metadata fragment having the values of the multi-key. The identification information on the metadata fragment having the values of the multi-key includes a 'target_container' segment storing therein the container identifier information (container_id) of the container in which the metadata fragment is stored and a 'target_handle' segment storing therein the fragment data identifier information (handle_value).

[158] The metadata structure will be more easily understood through FIG. 10, which illustrates the index information.

[159] FIG. 10 shows the multi-key index list (key_index_list) section comprising the multi-keys for Service Id and Published Time. The upper node of the metadata fragment including the multi-keys related to Service Id and Published Time is 'BroadcastEvent' 310 as indicated by the shaded region in FIG. 3. Therefore, the XPath of '/TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent' for the 'BroadcastEvent' fragment is stored in the 'fragment_xpath_ptr' segment 111, and the XPaths of the multi-keys of the Service Id and the Published Time for the 'BroadcastEvent' fragment, which are '@ServiceId' 311a, and 'EventDescription/PublishedTime' 311b, are stored in the 'key_descriptor' segment 112.

[160] This metadata stream structure allows searches for and access to the metadata fragments to be conducted efficiently, when searches are conducted based on more than one conditions, i.e., compound condition searches are conducted.

[161] Although the multi-keys for Service Id and Published Time are referred to in this present embodiment by way of example, a variety of the multi-keys may also be employed in combination. For example, multi-keys for start and end times of a program in connection with a broadcast schedule, multi-keys for family and given names of a person (actor, director, or the like) involved in the program and so on can be used.

[162] Where the multi-keys for the start and end times of the program in connection with the broadcast schedule are used, an upper node of a metadata fragment including the multi-keys for the start and end times of the program can be 'Schedule' (not shown). Therefore, the XPath, '/TVAMain/ ProgramDescription/ProgramLocationTable/Schedule,' for the 'Schedule' fragment can be stored in the 'fragment_xpath_ptr' segment 111, and the XPaths, '@start' and '@end', of the multi-keys of the start and end times of the program for the 'Schedule' fragment can be stored in the 'key_descriptor' segment 112.

[163] Where the multi-keys for family and given names of a person (actor, director, or the like) involved in the program are used, an upper node of a metadata fragment including the multi-keys for the family and given names of the person (actor, director, or the like) can be 'PersonName' (not shown), and therefore, the XPath, 'TVAMain/ProgramDescription/CreditsInformation

Table/PersonName,' for the 'PersonName' fragment can be stored in the 'fragment_xpath_ptr' segment 111, and the XPaths, 'FamilyName' and 'GivenName', of the multi-keys for the family and given names of a person in the program for the 'PersonName' fragment can be stored in the 'key_descriptor' segment 112.

[164] FIG. 11 depicts a method of providing an index of metadata having a structure according to an embodiment of the present invention. As depicted, the index of metadata can be generated by a provider 200 providing audio/visual signals.

[165] A number of information on contents, that is, metadata is processed in a unit of fragment as described above (S100). Multi-keys are provided by combination of keys related to information on the fragments, for example, 'Service ID' and 'Published Time' (S200). Then, sub key index (sub_key_index) sections 114a and 114b are provided by multi-keys as provided in S200, in other words, by keys forming the multi-keys (S300), wherein values of the multi-keys divided on the basis of predetermined ranges are included in the sub key index (sub_key_index) sections 114a and 114b, and metadata fragment identification information including values of the multi-keys (that is, container identifier information (container_id) and fragment data identifier information (handle_value) respectively stored in the 'target_container' segment and the 'target_handle' segment shown in FIG. 9) is also included in the sub key index (sub_key_index) section 114a and 114b.

[166] Subsequently, a key index (key_index) section 120 containing therein the representative key value representing the values of the multi-key divided on the basis of a range as predetermined is provided (S400). For example, representative key values ('509/10:00' 113a, 519/10:00' 113b, etc.) representing the predetermined ranges (500~509/09:10~10:00 114a, 510~519/09:10~10:00 114b, etc.) of the values of the multi-key for the Service ID/Published Time as combined are included therein. In this embodiment, the Service ID has an upper order of priority over the Published Time. In the key index (key_index) section 120 is included identification information on the sub key index (sub_key_index) sections 114a and 114b storing therein values of the multi-keys provided on the basis of a range as predetermined (that is, container identifier information (container-id) of the container in which the sub key index (sub_key_index) section of FIG. 9 is stored, and sub key index identifier information).

[167] Meanwhile, a key index list (key_index-list) section 110 in which multi-key information, that is, location information of a metadata fragment to which each field constituting the provided multi-key belongs and location information of each field within the metadata fragment is arranged on a multi-key basis, is provided (S500). For example, where keys of the 'Service ID' and the 'Published Time' are in combination, the multi-key information of combined 'Service ID' and 'Published Time,' such as the XPath of a target metadata fragment for indexing(/TVAMain/ProgramDescription/Program

LocationTable/BroadcastEvent) and XPath of a multi-key for the metadata fragment (XPath '@ServiceID' of the Service ID and the XPath 'EventDescription/PublishedTime' of the Published Time) are included in the key index list (key_index_list) section 110.

[168] The above steps may be processed in reverse order in other embodiments of the present invention. Further, a step of providing a key index (key_index) section 120 including representative key values (S400) or a step of providing a key index list (key_index_list) section (S500) may be deleted depending on some embodiments of the present invention.

[169] Hereinafter, a searching method of obtaining metadata meeting more than one search condition by use of the multi-key index structure according to an embodiment of the present invention described above will be described with reference to FIG. 12.

[170] First, conditions for a search are inputted by the user (S1100). A value of a multi-key satisfying the search conditions as inputted is searched from the metadata index (S1200). A concerned metadata fragment is extracted by use of identification information of the metadata fragment corresponding to the value of the multi-key by use of the searched value of the multi-key (S1300). Through these steps, the metadata satisfying the search conditions is extracted. In the search conditions inputted by the user, fields and a value of a field to be searched are included.

[171] The step of searching for the value of the multi-key (S1200) comprises steps of determining location information of the metadata fragment to which fields of the inputted search conditions belong and location information of the fields within the metadata fragment (S1210), searching for a multi-key consisting of fields having location information identical to the location information determined above, in the key index list (key_index_list) section 110 by use of the determined location information, and searching for the key index (key_index) section 120 relative to the searched multi-key (S1220), searching for a representative key value composed of values of the fields inputted as search conditions in the key index (key_index) section 120, and searching for sub key index (sub_key_index) sections 114a and 114b including the values of the multi-key in the range indicated by the representative key value searched above (S1230), and searching for the value of the multi-key satisfying the search conditions in the sub key index (sub_key_index) sections 114a and 114b searched above (S1240).

[172] In the above steps S1220, S1230 and S1300, the steps of searching for the key index (key_index) section 120, sub key index (sub_key_index) section, and extracting the metadata fragment are respectively performed by use of identification information of the key index (key_index) section 120, identification information of the sub key index (sub_key_index) section and identification information of the metadata fragment.

[173] The searching method as shown in FIG. 12 may be employed in searching the Service ID and the Published Time described with reference to FIG. 10 in the following manner:

[174] Where a user inputs search conditions of the Service ID '507~514' and the Published Time '9:30~10:00'(S1100), location information of a concerned metadata fragment is determined from fields in combination of the Service ID in the range of '507~514' and the Published Time in the range of '9:30~10:00' and location information of the fields within the metadata fragment is determined (S1210).

[175] The Service ID and the Published Time inputted as search conditions respectively have '@ServiceId' and 'EventDescription/PublishedTime' as location information within the metadata fragment. On this basis, the location information of the concerned metadata fragment as an attribute of the concerned fragment, that is, the XPath is determined (S1210).

[176] To sum up, we can obtain the following from the above steps:

[177] XPath of the fragment:

[178] /TVAMain/ProgramDescription/ProgramLocationTable/BroadcastEvent

[179] - XPath of the Service Id: @ServiceId,

[180] - XPath of the Published Time: EventDescription/PublishedTime

[181] - Value of of the Service Id: 507<= ServiceId <= 514,

[182] - Value of the Published Time: 9:30 <= EventDescription/ PublishedTime <= 10:00.

[183] Subsequently, a multi-key corresponding to the XPath 111 of the metadata fragment and the XPath 112 of the Service Id/Published Time is searched in the key index list (key_index_list) section 110, and the identification information on the key index (key_index) section 120 including the searched multi-key is extracted (S1220). In the present embodiment, the Service Id is higher in priority than the Published Time. The representative key values of '509/10:00' 113a and '519/10:00' 113b, i.e., the representative key values indicating the ranges (500-509/09:10-10:00 114a, 510-519/09:10-10:00 114b) of values of the multi-key to which the values of the multi-key (507-514/09:30-10:00) corresponding to the search conditions belong, are searched from in key index (key_index) section 120 having the extracted identification information and the identification information on the sub key index (sub_key_index) sections 114a and 114b having the representative values is extracted (S1230). Values of the multi-keys, having values of keys '507/09:30,' '507/09:40,"... '509/10:00' and '510/09:30,' '510/09:40'... '514/10:00,' corresponding to the values of the multi-key (507~514/09:30~10:00) corresponding to the search conditions are searched from the sub key index (sub_key_index) sections 114a and 114b having the extracted identification information (S1240).

[184] The identification information on the metadata fragment corresponding to the values of the multi-keys searched (the container identifier information (container_id) and the fragment data identifier information (handle_value) stored in the 'target_container' segment and the 'target_handle' segment, respectively) is extracted and the relevant metadata fragment is then extracted by use of the extracted identification information (S1300).

[185] FIG. 13 shows an apparatus for searching for metadata according to an embodiment of the present invention. The searching apparatus in the present invention is an apparatus performing a method of searching for the metadata according to an embodiment of the present invention described above with reference to FIG. 12.

[186] The searching apparatus comprises an input unit 1100 allowing a user to input search conditions therewith, a receiving unit 1200 receiving metadata on contents or an index of the metadata, a storage unit 1300 storing therein received contents, metadata on the contents or an index of the metadata, a control unit 1400 searching for a value of the multi-key corresponding to the input search conditions from the input unit 1100 from the metadata index, and extracting the concerned metadata by use of the value of the multi-key as searched, and an output unit 1500 outputting the search result of the control unit 1400.

[187] The control unit 1400 compares the search conditions inputted from the input unit 1100 with the value of the multi-key included in the metadata index stored in the storage unit.

[188] Among the steps of searching for values of the multi-key according to one embodiment of the present invention, a step of searching for a multi-key corresponding to the input search conditions (S1200), or a step of extracting the concerned fragment by use of identification information of the fragment corresponding to the searched multi-key will be understood with reference to the description made above in connection with FIG. 12.

[189] According to the present invention, there is provided an index structure of metadata allowing more efficient search for and access to information on contents, a method of providing the metadata index having the structure, and a method and an apparatus for searching for metadata using the metadata index.

[190] As described above, the present invention enables simultaneous searches by compound conditions for TV Anytime metadata. Where searches by compound conditions for the TV Anytime metadata are conducted, overhead to a searching apparatus is decreased, thereby allowing search time to be shortened and the searching apparatus to be increased in terms of efficiency.

[191] Although the present invention has been described in connection with the preferred embodiment shown in the drawings, it is only illustrative. It will be understood to those skilled in the art that various modifications and